

Introduction to Data Science

Data Mining for Business Analytics

BRIAN D'ALESSANDRO

ADJUNCT PROFESSOR, NYU

FALL 2018

Fine Print: these slides are, and always will be a work in progress. The material presented herein is original, inspired, or borrowed from others' work. Where possible, attribution and acknowledgement will be made to content's original source. Do not distribute, except for as needed as a pedagogical tool in the subject of Data Science.

DATA PREP FOR MODELING

DATA MUNGING

Before any analysis we need to carefully craft our matrix of instances (rows), features and a label.

Set of considerations = {Instances, Features, Labels}

$$\text{Data} = \begin{bmatrix} X_{N \times K} & Y_{N \times 1} \end{bmatrix}$$

DEFINE AN INSTANCE OF THE DATA

What is the instance space for your problem?

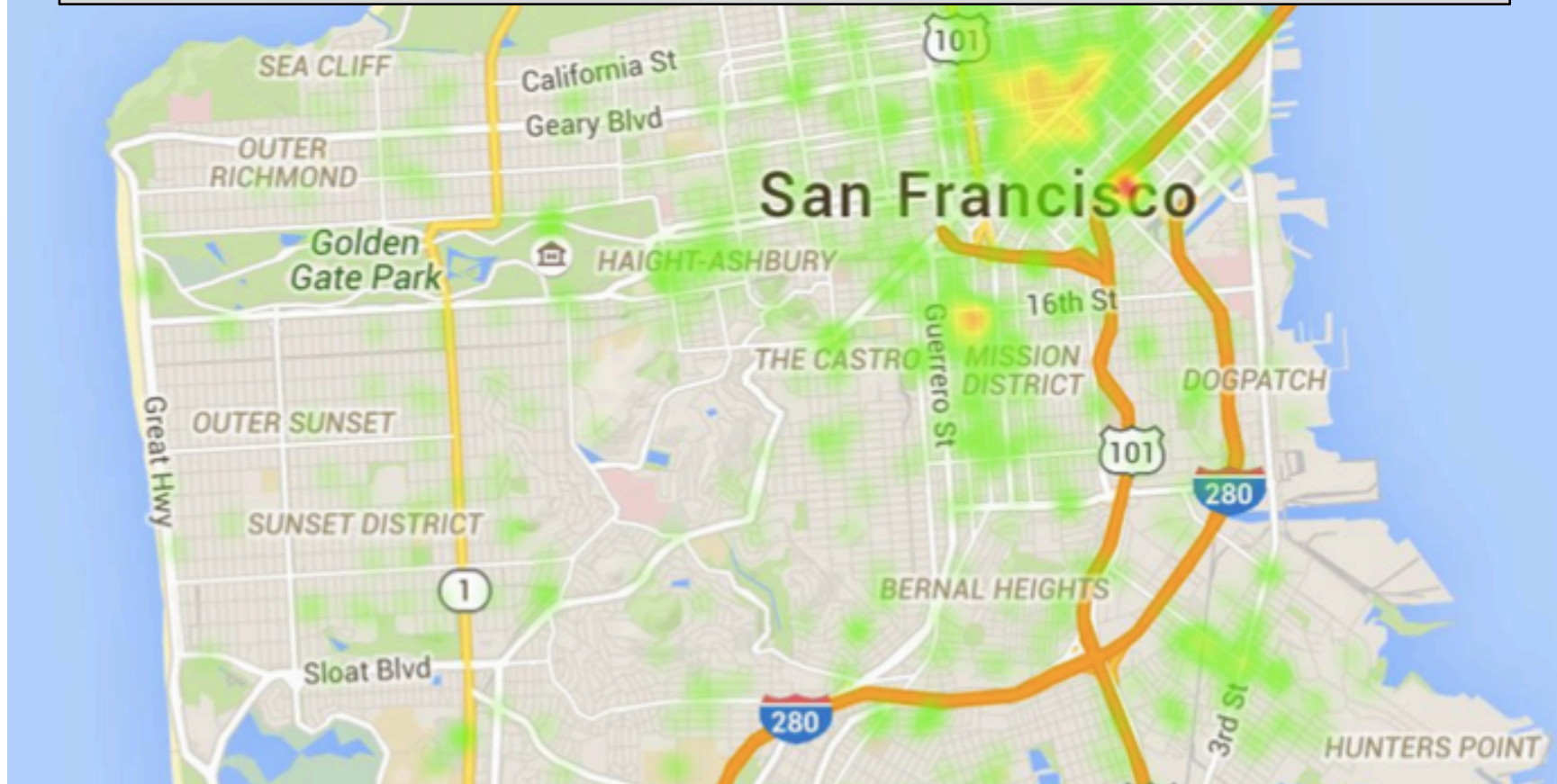
This sounds like it should be trivial, but it does require careful thinking in certain problems!

1. Who or what should be sampled? Are positives and negatives drawn from the same population or process? Some times we observe only positives and have to find appropriate negatives.
 - *Examples of same process/population:*
 - Kickstarter projects: funded/not funded
 - Ads served: clicked/not clicked
 - *Examples of not same process/population:*
 - People with a disease at a clinic (pos) + random hospital patients (neg)
 - People visiting your site (pos) + random internet browsers (neg)
2. Are the instances independent of each other?
 - Geo-spatial data
 - Time series data
 - Pairwise instances (social networks, search)

DEFINE AN INSTANCE OF THE DATA

Example 1: SF Crime data. There is no automatically well defined instance. We have crime data on a map. We need to discretize across time and space to make geo-time instances.

Note: these instances won't be independent – there will both be spatial and temporal correlations in the data.

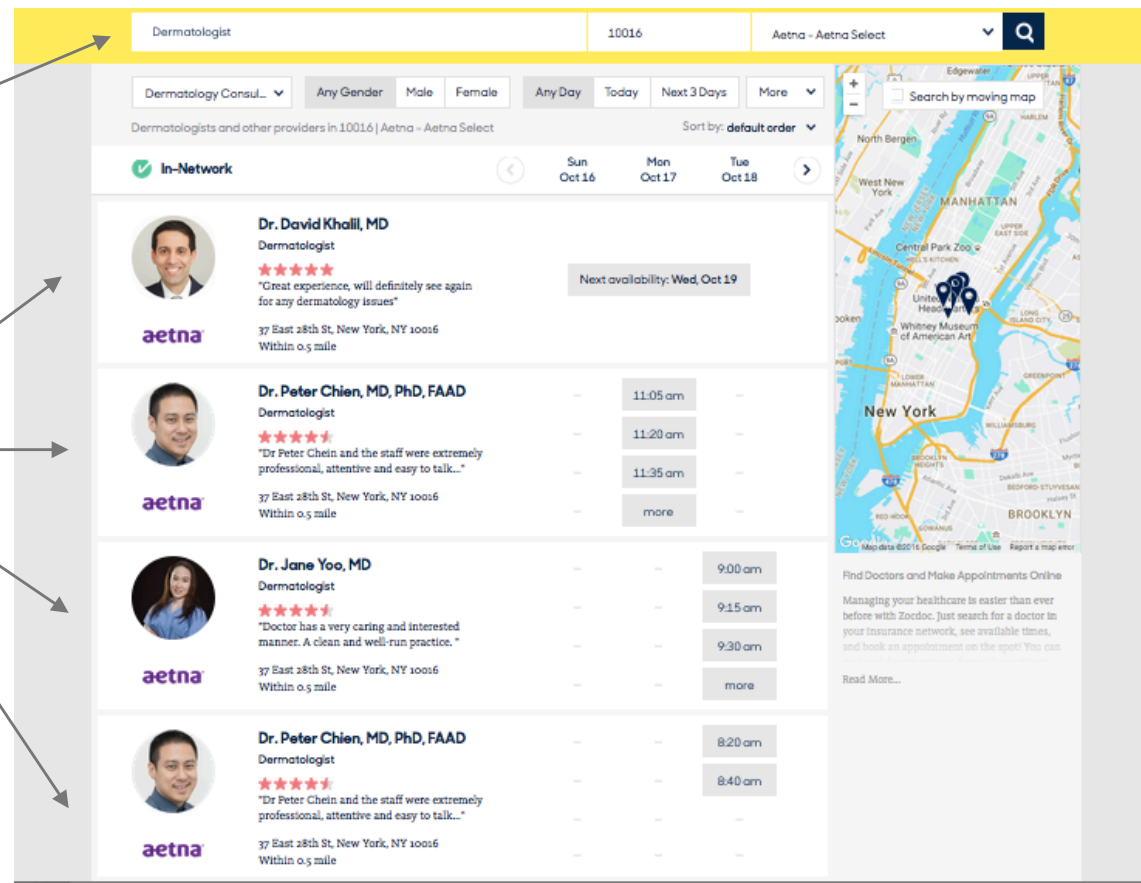


DEFINE AN INSTANCE OF THE DATA

Example 2: Search. Is a single search the instance? No, we need to define each search result – search query as an instance.

The user query defines a discrete search.

Each search result is an independent exposure, on which we observe a label of click or not.

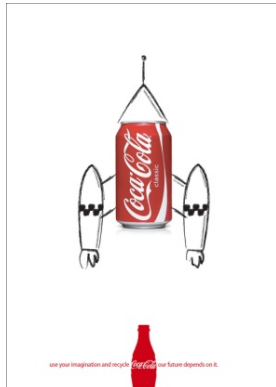


CHOOSE THE TARGET VARIABLE

This should be directly determined by the problem. I.e., what are you trying to predict?

Will someone click on an ad?:

$C=[No, Yes]$



Is this pill good for headaches?:

$C=[No, Yes]$

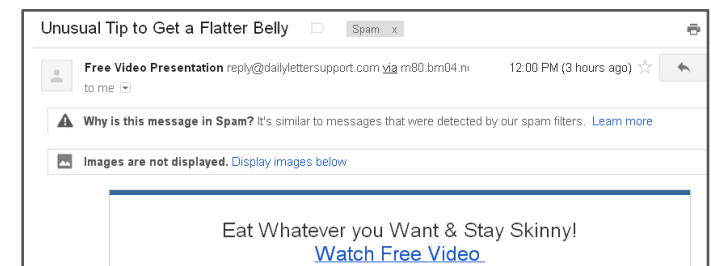


What number is this?:

$C=[0,1,2,3,4,5,6,7,8,9]$

7210414959
0690159784
9665407401
3134727121
1742351244

Is this e-mail spam?: $C=[No, Yes]$



What is this news article about?:

$C=[Politics, Sports, Finance ...]$



SOMETIMES YOU HAVE TO BE CREATIVE TO DEFINE A LABEL

Target variables aren't always obviously specified, or sometimes we transform the logical target variable to create a simpler problem.

- What do we want to model?
 - Search & Recommendation: Clicks on a recommendation vs purchases?
 - Newsfeeds: likes, comments, hovers?
- Yelp reviews: predict a positive review
 - $Y \in \{1, 2, 3, 4, 5\} \Rightarrow Y' = I(Y \geq 4)$
- Survival analysis (T = Time until we observe event E)
 - $Y' = I(T < k)$
- Proxy modeling (we don't observe someone buying a car online, but we observe them visiting the car dealer's web site)
 - $Y = I(\text{Buys Car} = \text{True}) \Rightarrow Y' = I(\text{Visits Web Site} = \text{True})$
 - Assumption: Y and Y' are highly correlated, by Y' more abundant

SAMPLING

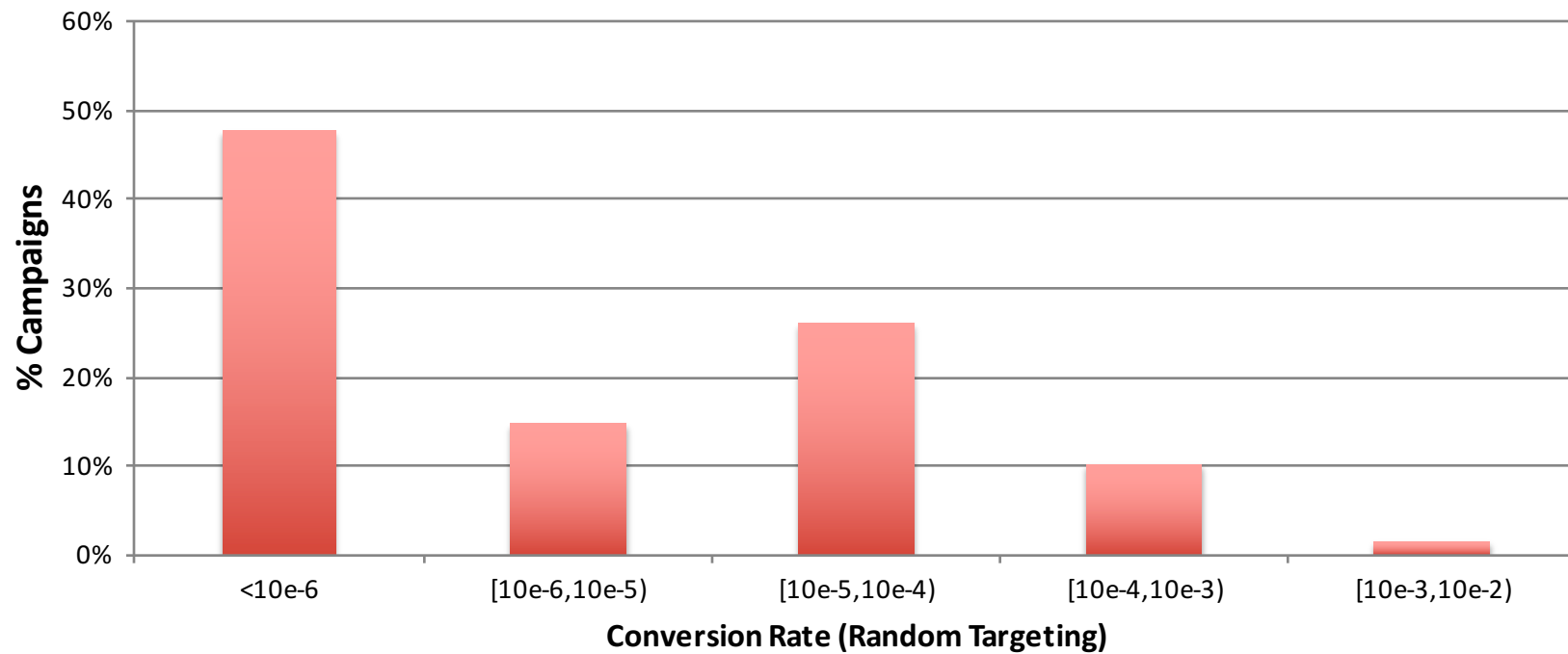
- Data is often big, and whether or not you need all of it is a case by case decision. Despite this, there are trends to guide you.
- The most obvious sampling strategy is to take everything you have.
- Two common sampling strategies used in practice:
 - ‘down-sampling.’
 - Take 100% of the minority class,
 - Take K% of the dominant class
 - ‘up – sampling’
 - Take 100% of the dominant class
 - Sample with replacement the minority class until equal

A CASE FOR DOWN-SAMPLING

Positive outcomes are often very rare, such that to get 1 positive outcome you need 1k-1MM negatives.

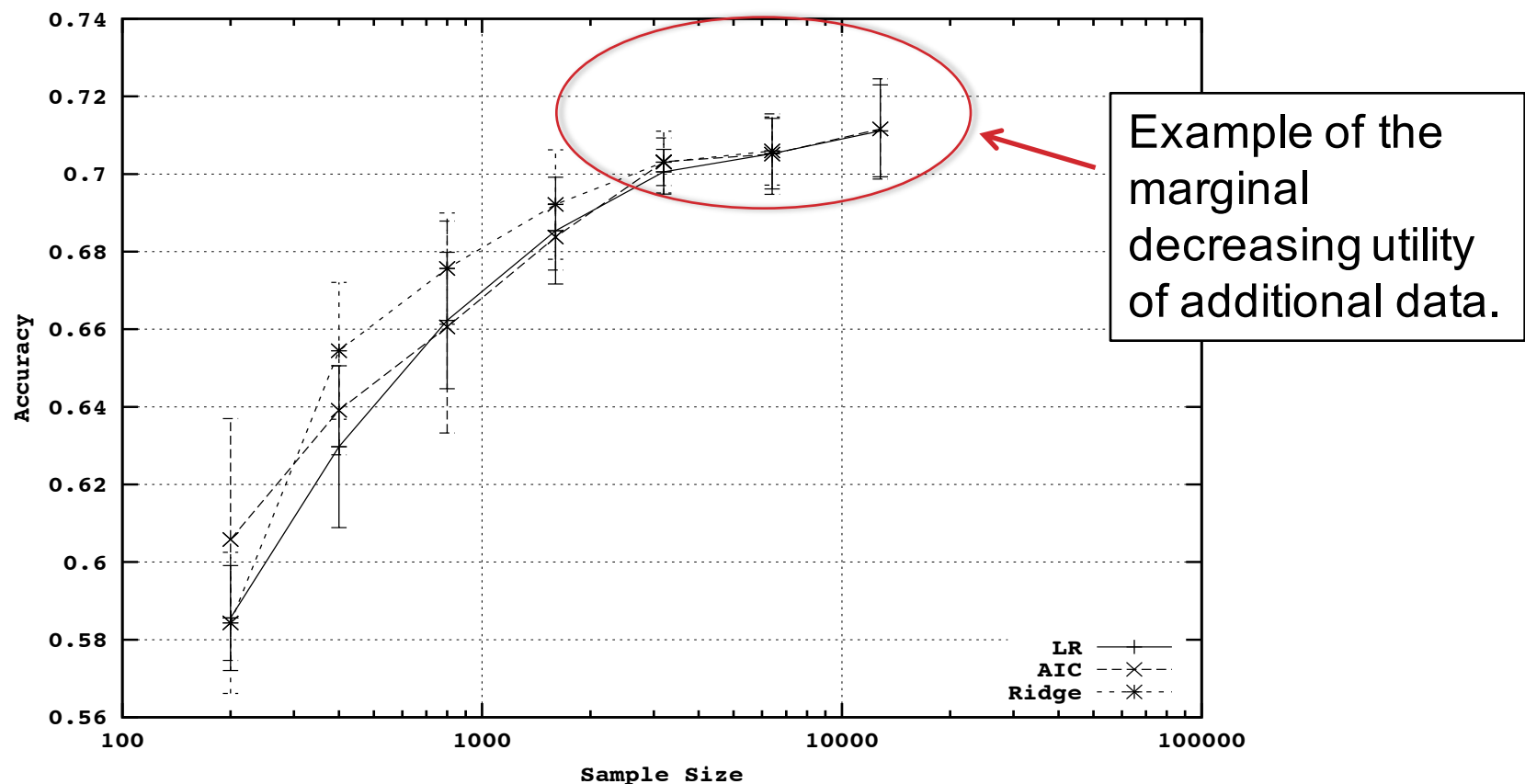
As N grows, data in the dominant class may have marginal decreasing utility. Thus, the cost of processing or storing it outweighs the benefit of using it.

Distribution of Conversion Rates for Display Ad Campaigns



SHOULD YOU DOWN SAMPLE?

This is largely an empirical question. Rule of thumb – less complex algorithms & models with information rich features require less data. Learning curves are a good way to visualize and measure the sample size – performance tradeoff

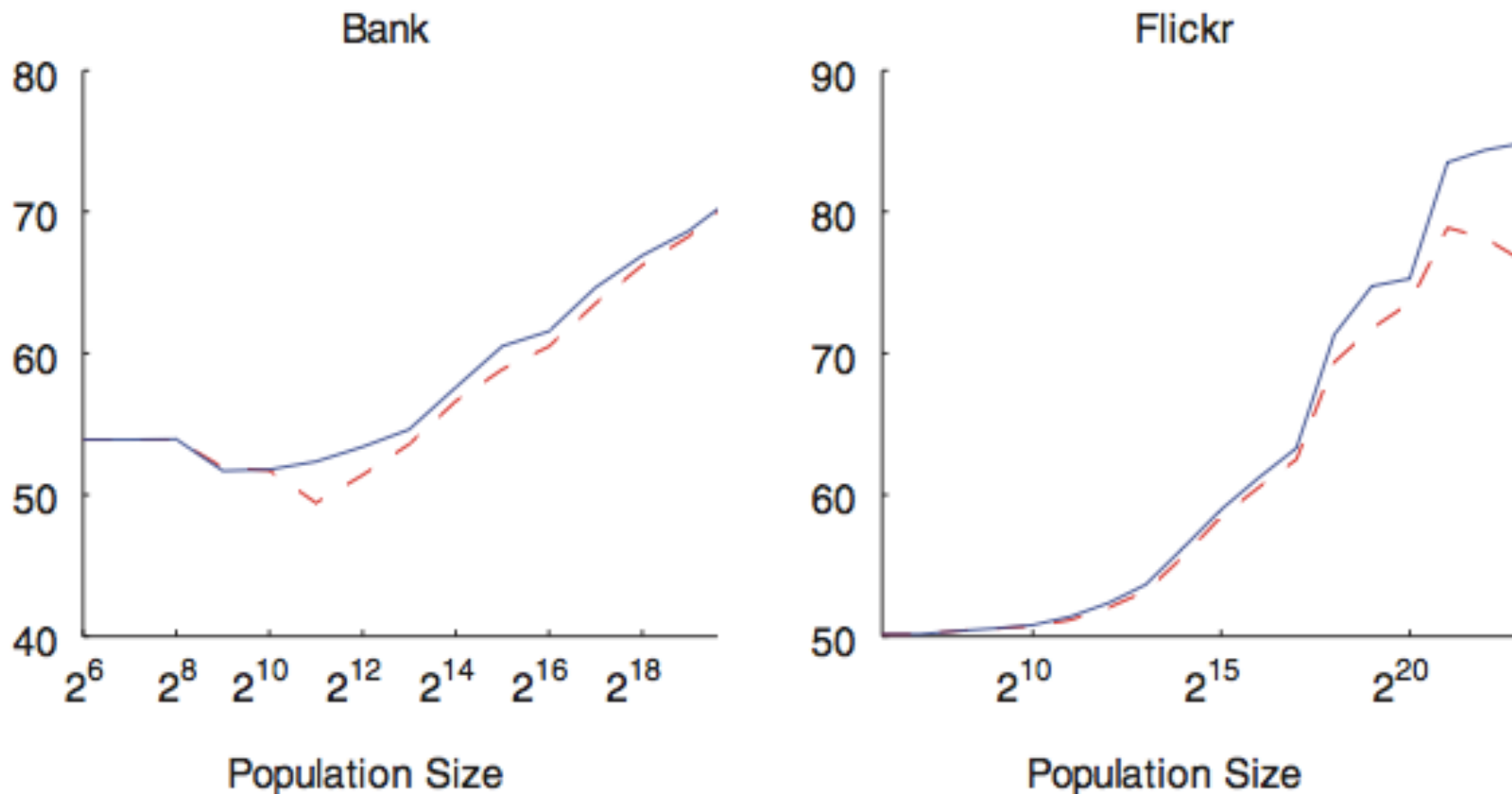


Source: Tree Induction vs. Logistic Regression, a Learning Curve Analysis
<http://pages.stern.nyu.edu/~fprovost/Papers/logtree.pdf>

NYU – Intro to Data Science
Copyright: Brian d'Alessandro, all rights reserved

ON THE OTHER HAND...

Certain problems where the data contains many sparse and uninformative features fare much better when more data is present.



Source: Predictive Modeling with Big Data: Is Bigger Really Better?
<http://online.liebertpub.com/doi/pdfplus/10.1089/big.2013.0037>

NYU – Intro to Data Science
Copyright: Brian d'Alessandro, all rights reserved

IF YOU DOWN SAMPLE.

1. Use Learning Curve analysis to justify down-sampling rate

2. Be aware of the effect on probability estimates.

- $P(Y)$ and $P(Y|X)$ changes when you down sample based on Y
- Weighting the down-sampled class by $1/k\%$ can correct for this
- Can adjust intercepts if applicable:

$$\hat{\beta}_0 - \ln \left[\left(\frac{1 - \tau}{\tau} \right) \left(\frac{\bar{y}}{1 - \bar{y}} \right) \right]$$

where τ is the base rate of the population and \hat{y} is the sample base rate.*

3. Be aware of the effect of base rate on evaluation metrics

- AUC is invariant to base rate
- Accuracy, precision and lift depend on the base rate

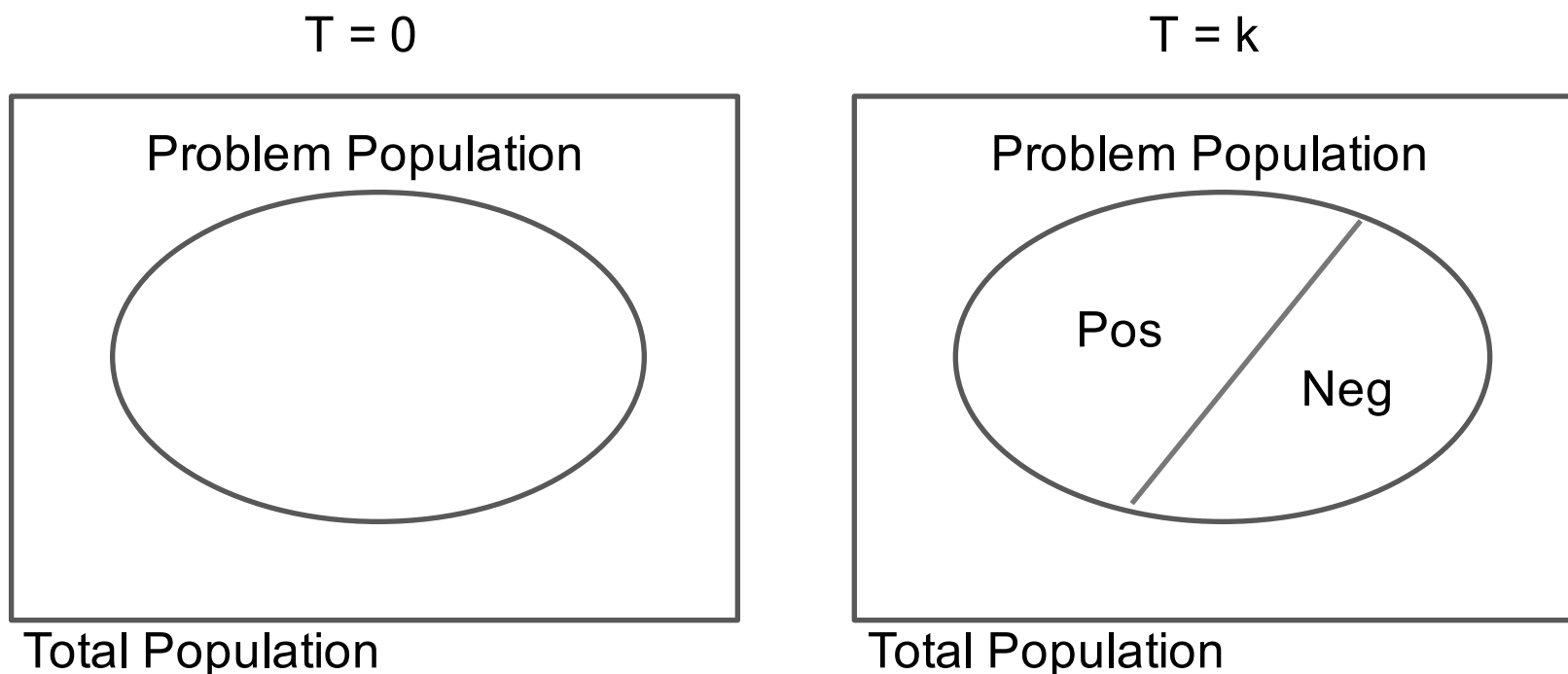
Source: Logistic Regression in Rare Events Data

<http://dash.harvard.edu/bitstream/handle/1/4125045/relogit%20rare%20events.pdf?sequence=2>

COUPLING LABELING W/ SAMPLING

Example labeling processes:

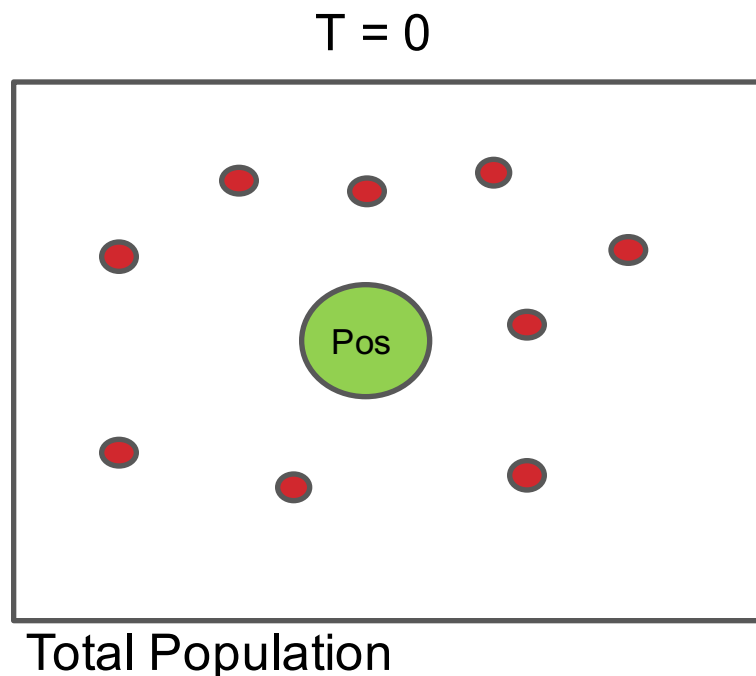
In many problems we can define a population of interest and then collect the labels through some intervention process or observation period. I.e., ad serving, credit risk, fraud,



COUPLING LABELING W/ SAMPLING

Example labeling processes:

In some problems we sample two populations, labeling each either pos/neg (called case-control study design in medical studies). Retrospective as opposed to prospective.



Problem: Look-alike modeling – create an audience that is similar in activity and composition to a target audience. (FB, Display Advertising)

Pos: All people that have purchased a particular product or have been to a particular site.

Neg: Randomly selected people who are not in the Pos population.

See: http://archive.nyu.edu/bitstream/2451/31708/2/Provost%201_2013.pdf

DO APPROPRIATE DATA TRANSFORMATIONS

This is also called Feature Engineering, and is a great skill to develop.

Types of Feature Engineering

- Binning
- Non-linear transformations
- Domain knowledge feature extraction

BINNING (ONE HOT ENCODING)

Taking categorical data with K values (also called a factor) and projecting them to K-1 binary features (also called 'dummy indicators').

Data can be numeric or categorical



If there are 4 categories/ranges, we have 3 binary variables.

User	Income	IncomeGrp
1	\$65,500	[61k,80k]
2	\$81,041	[81k,100k]
3	\$38,346	[21k,40k]
4	\$47,072	[41,60k]
5	\$30,812	[21k,40k]
6	\$21,618	[21k,40k]
7	\$97,872	[81k,100k]



User	inc_41_60	inc_61_80	inc_81_100
1	0	1	0
2	0	0	1
3	0	0	0
4	1	0	0
5	0	0	0
6	0	0	0
7	0	0	1

1. Numeric data can be binned based on a pre-defined range.
2. Categorical data can be binned based on category value.
3. If you make K indicators and not K-1, your X matrix will be degenerate/singular because because the Kth indicator will be a linear combination of the K-1 previous indicators. Many algorithms don't like non-invertible matrices.

NON-LINEAR TRANSFORMATIONS

Let's say you want/have to build a linear model, so you need to express your dependency as:

$$E[Y|X] = \alpha^0 + \beta^0 * X$$

But the phenomenon/data you are modeling is actually non-linear, so to get a better fit you want:

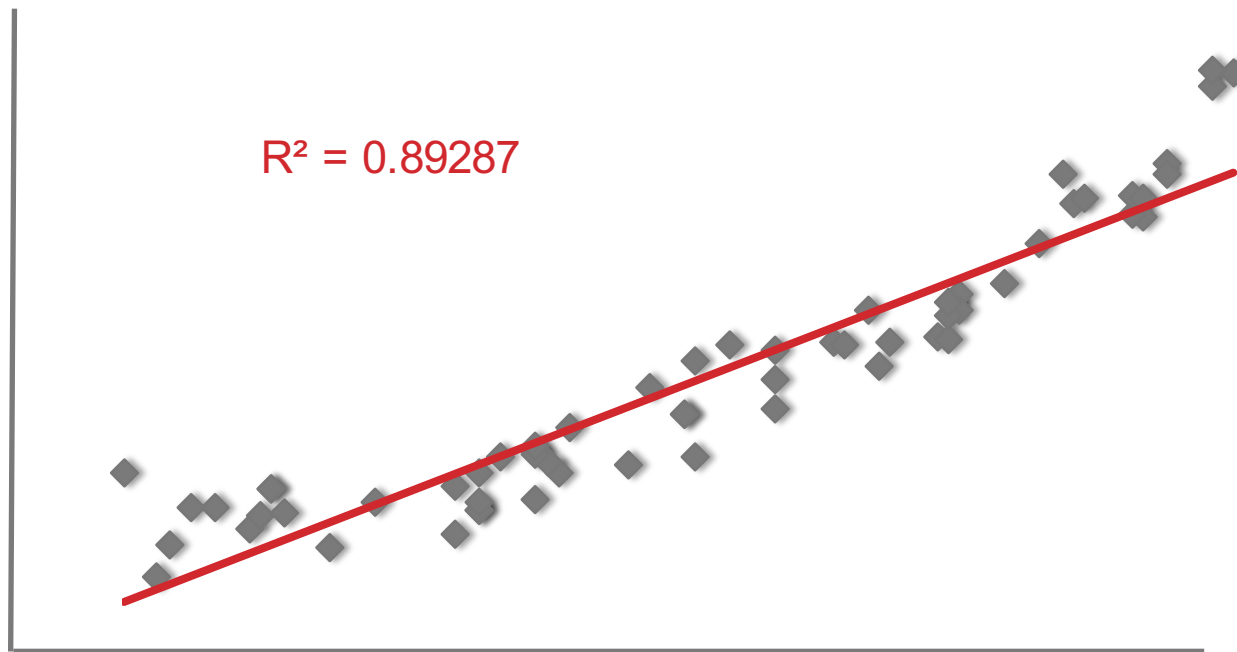
$$E[Y|X] = \alpha^1 + \beta^1 * g(X)$$

In lieu of using more sophisticated non-linear algorithms (i.e., Random Forest, Neural Network), you can make clever feature transformations to get a non-linear fit out of a linear algorithm.

EXAMPLE – NON-LINEAR TRANSFORM

In this example we have a regression problem with the following model specification: $E[Y|X] = \alpha + \beta * X$

The R^2 is pretty good, but visually, we can see a potentially better fit. What transformation might fit this data better?

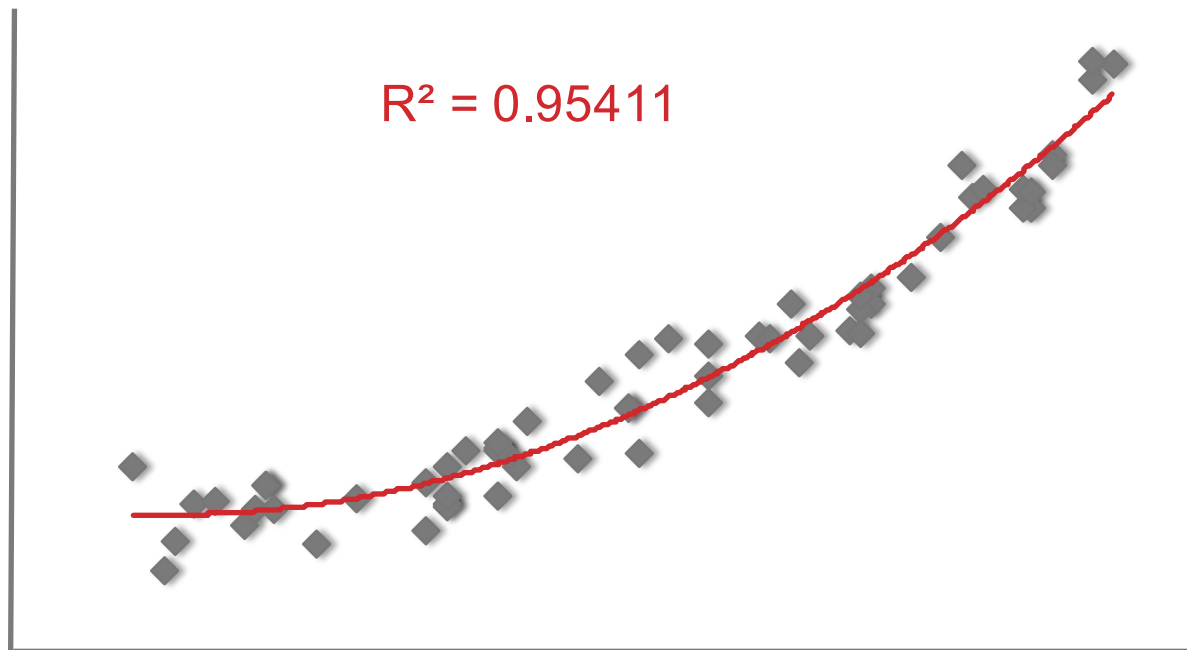


EXAMPLE – NON-LINEAR TRANSFORM

Lets use the transformation $g(X) = \langle X^2, X \rangle$

We can then fit the model: $E[Y|X] = \alpha + \beta^*g(X) = \alpha + \beta_1X^2 + \beta_2X$

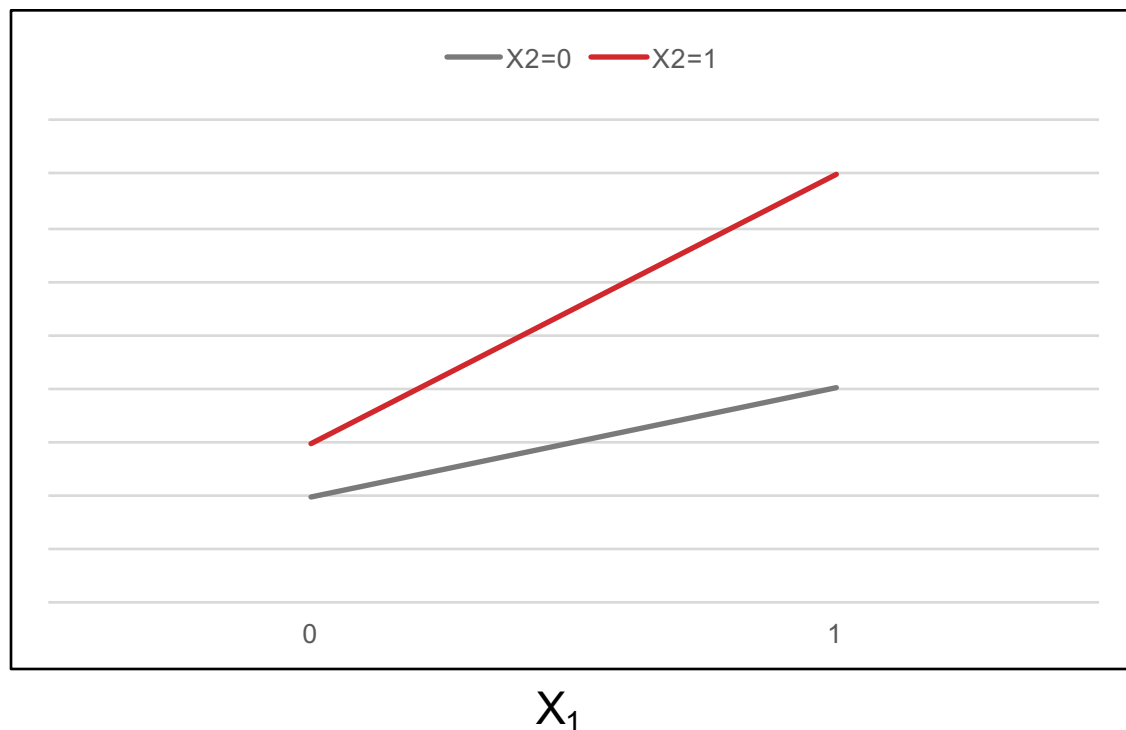
We can often realize a non-trivial increase in performance by making such transformations prior to modeling.



INTERACTION TERMS

When $dE[Y|X_1] / dX_1$ depends on X_2 , we say X_1 and X_2 show an interaction effect. In many cases we have to construct explicit features to capture them.

Sample model: $Y = a + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_1 X_2$



Logical:

$X_1_X2_int = (X_1 = 1 \text{ and } X_2 = 1)$

Mathematical:

$X_1_X2_int = X_1 * X_2$

NON-LINEAR TRANSFORMATIONS IN D-DIMENSIONS

A polynomial transformation in multiple dimensions captures both the interaction effects and the polynomial transformations of each feature.

$$X = \langle X_1, X_2, \dots, X_k \rangle$$

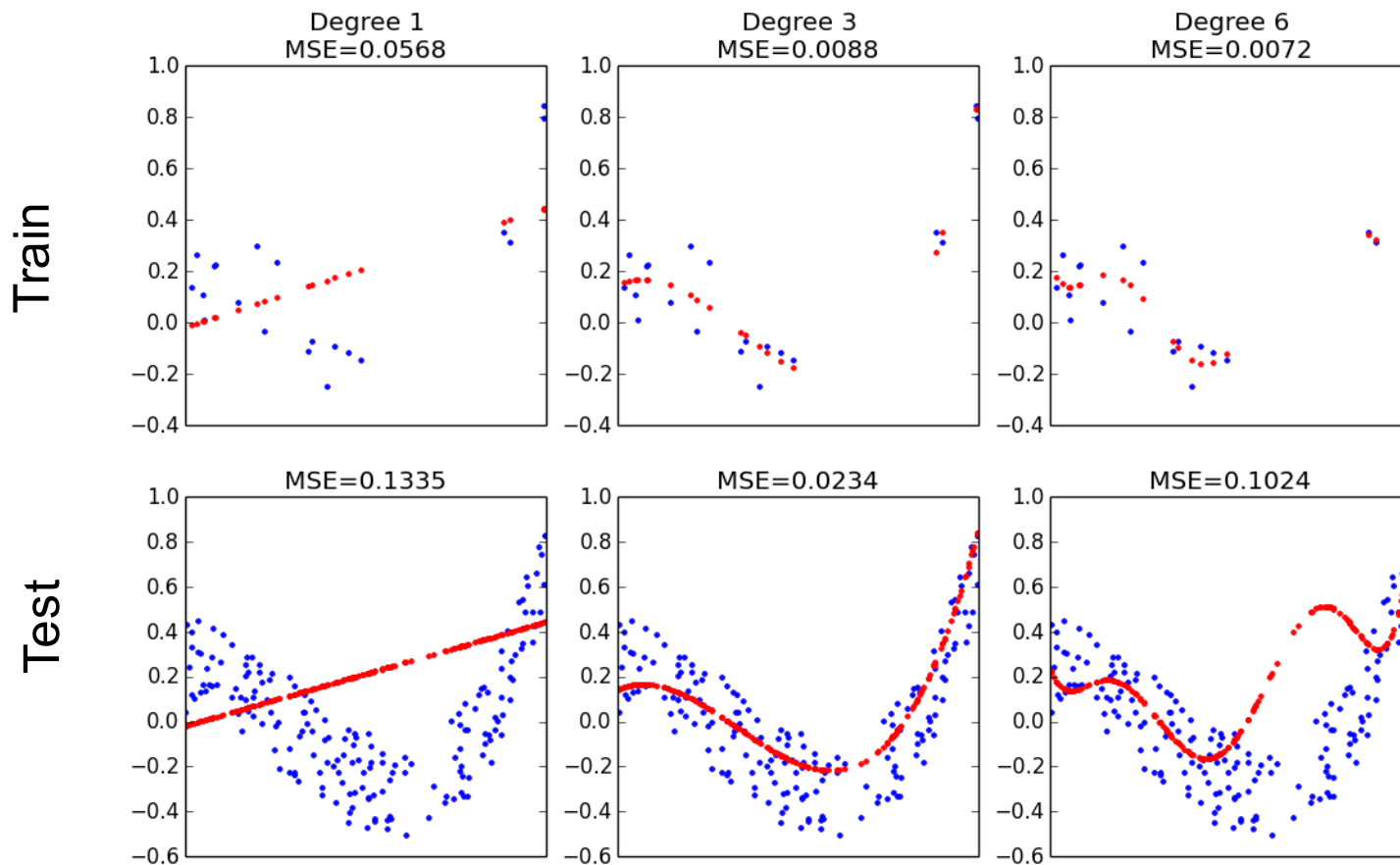
$$E[Y|X] = \alpha + \beta^* g_d(X); g_d(X) \Rightarrow (X+1)^d$$

$$\text{e.g., } g_2(X) \Rightarrow \langle 1, X_1, X_2, X_1 * X_2, X_1^2, X_2^2 \rangle$$

The use of Kernels in SVM's can generate these transformations implicitly, so no new explicit feature generation would be needed.

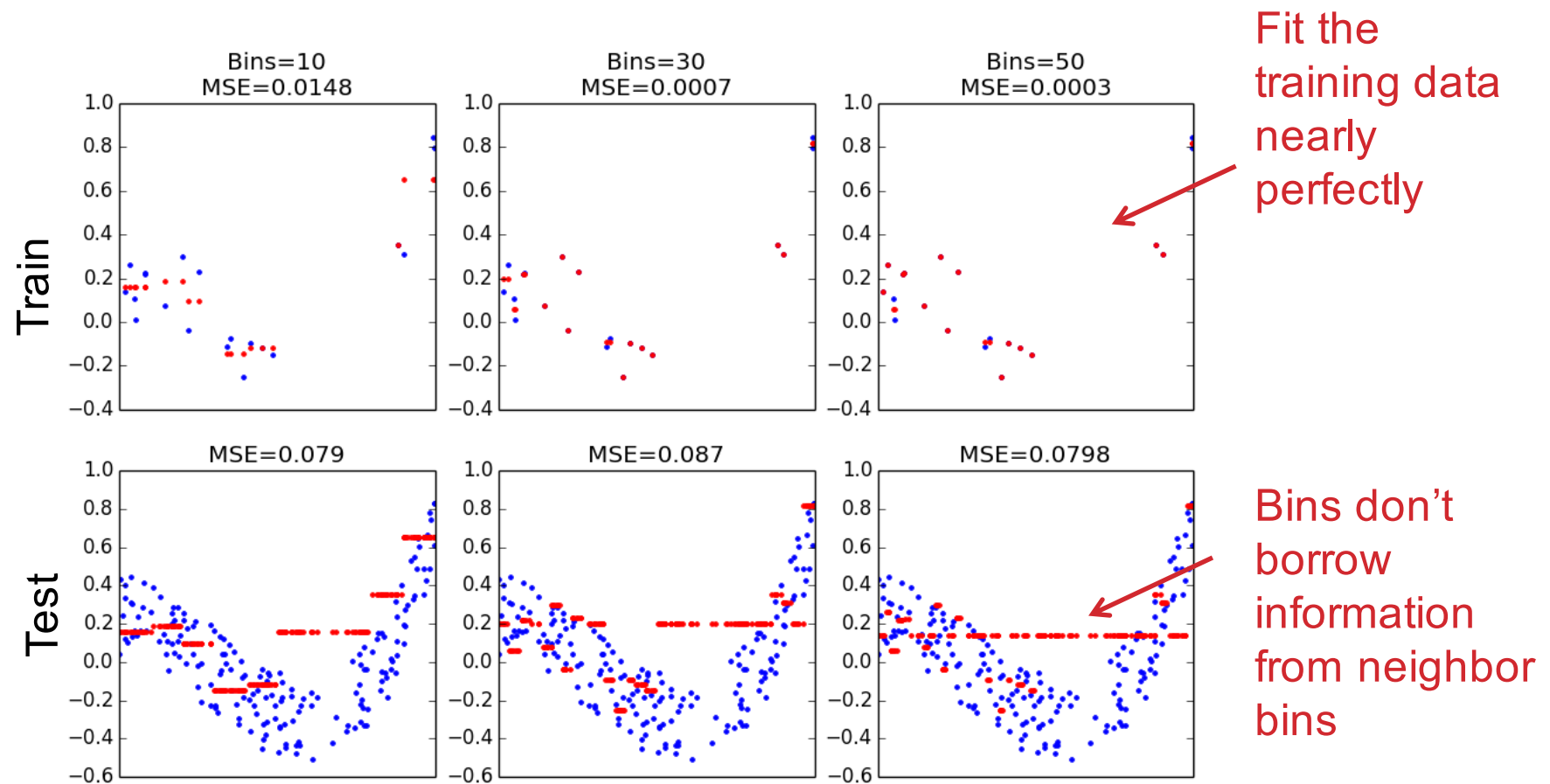
NL TRANSFORMATION: NOISY ENVIRONMENT

If $g(X)$ is a polynomial expansion of X (i.e., $g^d(x) = \langle X^d, \dots, X^2, X \rangle$), we can get a great fit by choosing the right degree of expansion. Too low and we get generally bad MSE everywhere...too high and we overfit.



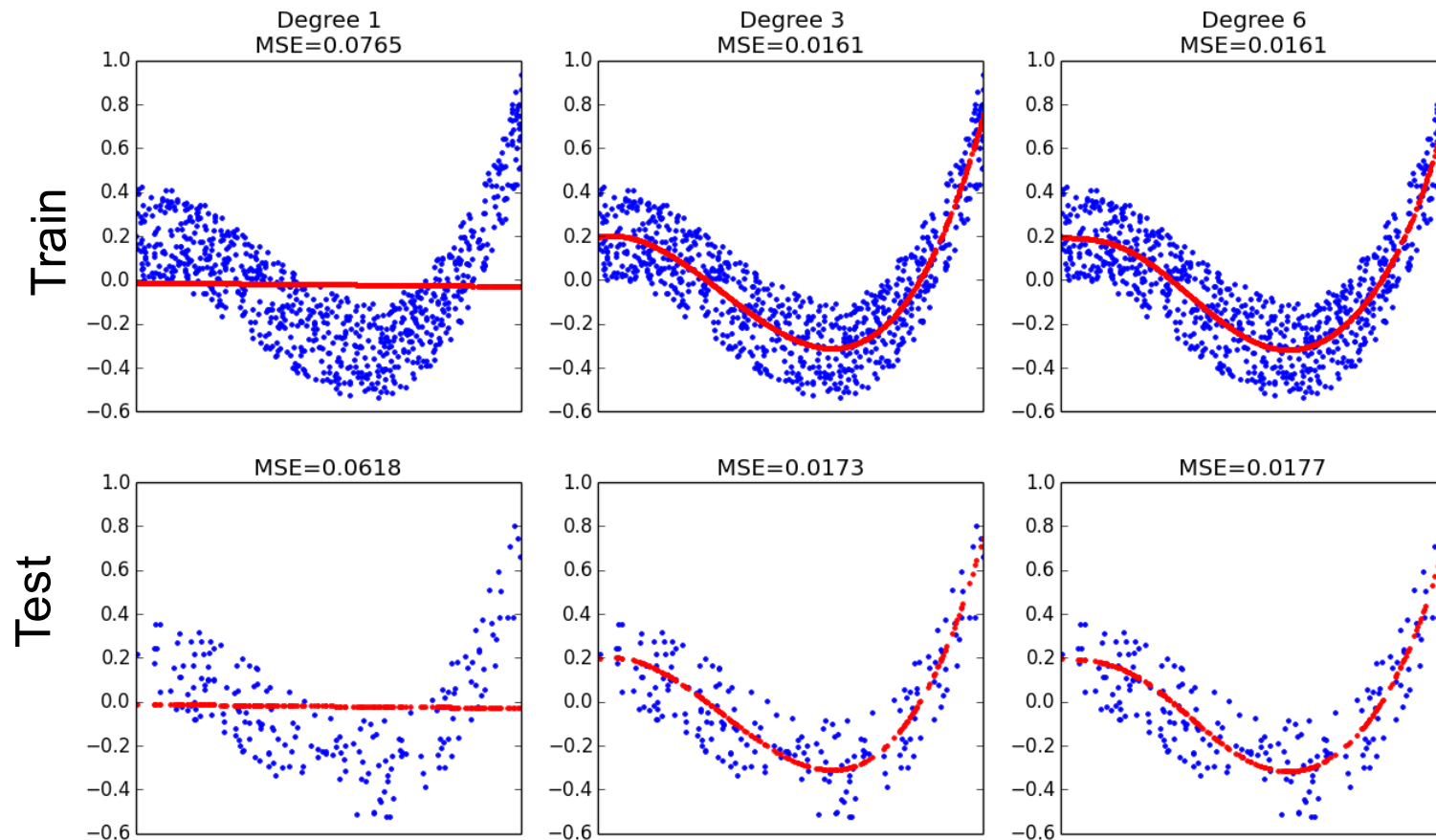
BINNING: NOISY ENVIRONMENT

Binning a numeric feature also needs to be done carefully. Too few bins and you get a poor fit...too many bins and you overfit terribly.



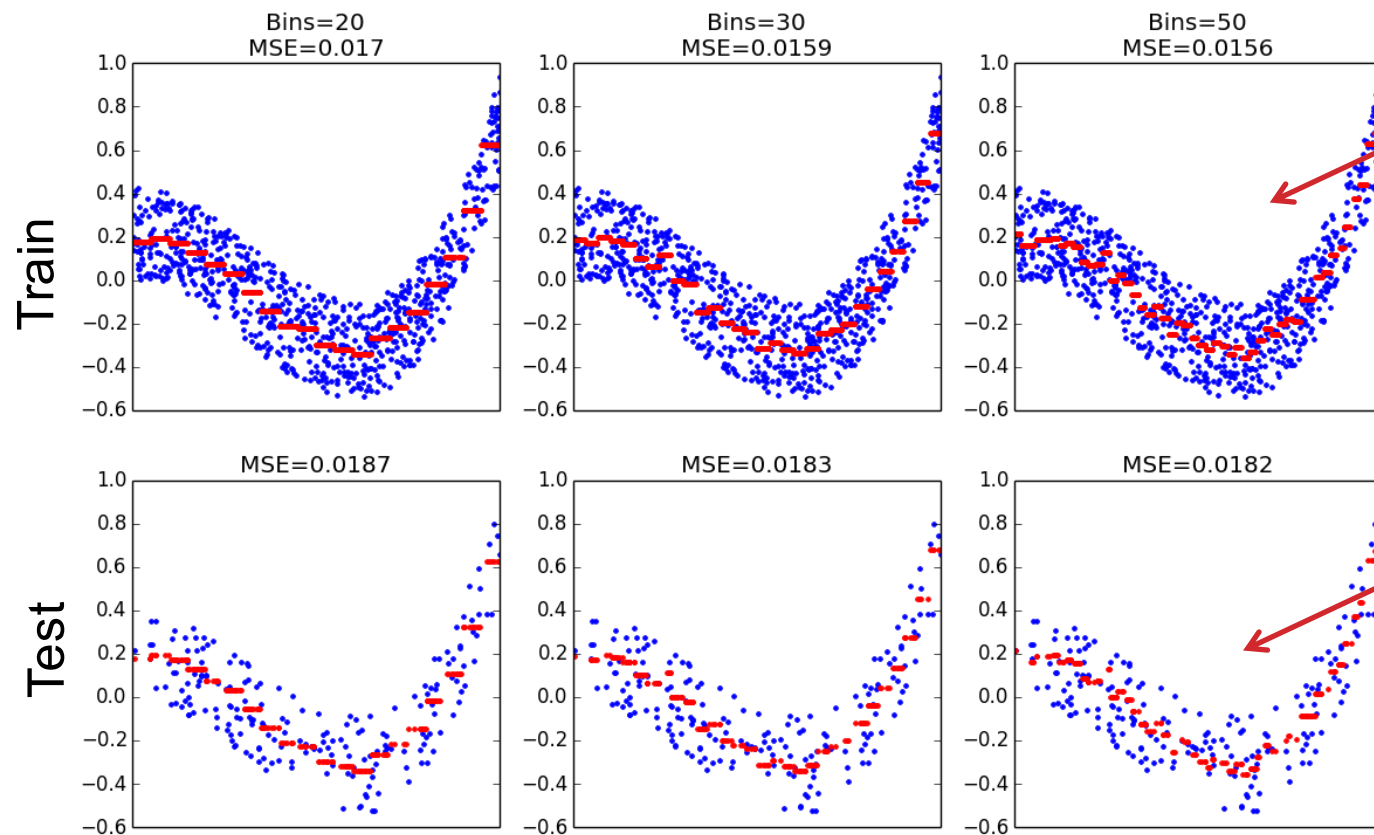
NL TRANSFORMATION: INFORMATION RICH ENVIRONMENT

When we have much more training data to fit, the highest order polynomial doesn't help much, but is less likely to overfit. The data was generated using a degree 3 polynomial, and our regression has effectively learned that.



BINNING: INFO RICH ENVIRONMENT

With more data we can support a higher number of bins. This method can be competitive with the polynomial fit, but it is still easy to over fit. Careful consideration and testing must be done to choose the appropriate bin size.



Binning can approximate any arbitrary curve.

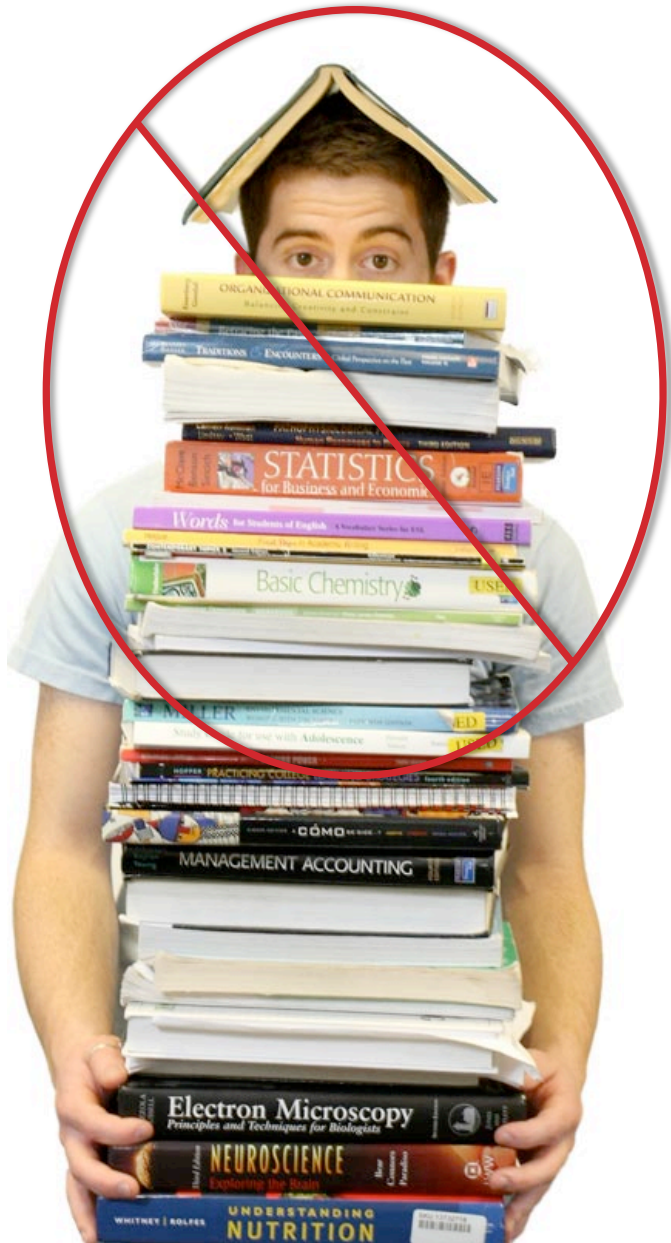
But it is always at a high risk of overfitting.

DOMAIN KNOWLEDGE FEATURE EXTRACTION

- Many industrial data collection processes store data in either primitive log formats or in schemas designed to support transactional systems.
- **These systems are usually built for speed, not analytics.**
- As a result, data scientists often need to design their own features to be used in analysis and modeling.



HOW TO DESIGN THE PERFECT FEATURE?



- Unfortunately there are usually no text books or theoretical systems teaching us how to design good features
- Intuition, creativity and knowledge of the application domain are needed.
- Data scientists should consult with business experts when doing this.

SOME GUIDELINES FOR DOMAIN KNOWLEDGE FEATURE ENGINEERING

Try to understand the physical mechanism at work

Not all processes you model have theoretical models that define them. Nonetheless, you can put yourself in the user's shoes. What are actions and events that you think might be correlated with the target variable.

Thoroughly review the underlying data system

The data is collected to support the product and transactional systems. These might lead clues as to what can or should be used as features.

Consult with business people

People with extensive knowledge of the business might have solid and experience driven intuition that can help you formulate a feature plan.

Put yourself in the user's shoes

What would you do before: buying something, clicking on an ad, defaulting on your credit card, friending someone on FB? Try to encapsulate this behavior in the form of a feature.

LEAKAGE

1. Target Variable Leakage - Having a feature that is caused by the outcome of the target variable.

Examples:

- using the fact that a user saw a “Thank You” page to predict that the user will purchase (hint: if the feature happens after the outcome, it’s probably not a legitimate feature)
- Positives and Negatives are stored in different log files, and each log uses a different range for the user id. The user id perfectly predicts the outcome!

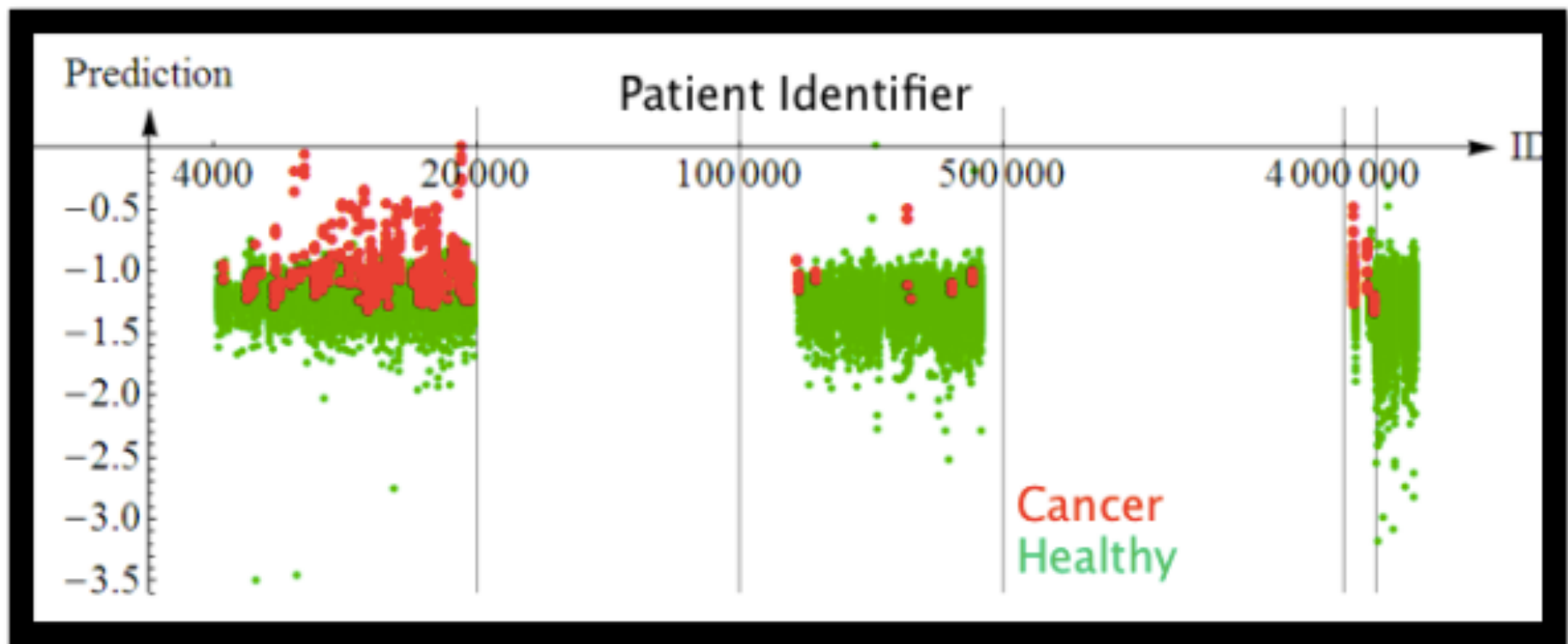
2. Training/Testing Leakage – Having records in the training set also appear in the test set

Model performance that is too good to be true is a good reason to suspect leakage!

LEAKAGE - EXAMPLE

Patients with Cancer were pulled from a different system and generally had a different distribution of IDs. But ID is a worthless artifact of the system, and not a true predictor of cancer.

Leakage is often caused by sloppy data prep!!!



From the KDD 2008 Cup Winning Solution, provided by Claudia Perlich.

LET'S BUILD A DATASET FOR TWITTER RECOMMENDATIONS

Who to follow

Twitter accounts suggested for you based on who you follow and more.

Search using a person's full name or @username

Search Twitter



Johan Ugander @jugander

I study social graphs and the structure of social data.
Present: MSR Redmond postdoc; Future: Stanford MS&E faculty, Fall 2015; Past: Cornell Applied Math PhD

Followed by josh attenbergh, jake hofman and Blake Shaw.



+ Follow



Moira Burke @grammarnerd

Data scientist at Facebook. Recent CMU grad. I study people and the internet.



+ Follow



Michael E. Driscoll @medriscoll

Founder + CEO @Metamarkets. Investor @DCVC. I ❤️ data, analytics, & visualization.

Followed by Randy Au, jake hofman and chris wiggins.



+ Follow











Mark Faridani @siah

Just finished a PhD at Berkeley. Silicon Valley Scientist. Machine Learning, HCI, Data Science and Statistics. Addicted to coffee and in love with @marjoo.





+ Follow

DATA INSTANCES

<u>User i</u>	<u>User j</u>	<u>Label</u>	<u>Feature</u>
		1	$\langle X_1, X_2, \dots, X_k \rangle$
		0	$\langle X_1, X_2, \dots, X_k \rangle$
		0	$\langle X_1, X_2, \dots, X_k \rangle$
		1	$\langle X_1, X_2, \dots, X_k \rangle$

FEATURE TYPES

In a dataset where instances are pairwise combinations of individuals, we can group features into sets that reflect individual characteristics as well as pairwise characteristics.

<u>User i</u>	<u>User j</u>	<u>Label</u>	<u>Features</u>
		1	$\langle X_1, X_2, \dots, X_k \rangle$

User_i features = <number of followers, interests, etc.>

User_j features = <number of followers, number of follows, interests, etc.>

Pairwise features = <number of friends in common, degree of separation, number common interests, etc.>

GENERATE NETWORK BASED FEATURES

